EXHIBIT 47

SPRING 2003 VOL.44 NO.3

MITSloan
Management Review

Georg von Krogh

Open-Source Software Development

REPRINT NUMBER 4432

RESEARCH BRIEF

Open-Source Software Development

An overview of new research on innovators' incentives and the innovation process. **by Georg von Krogh**

Open-source software development projects — Internet-based communities of software developers who voluntarily collaborate in order to develop software that they or their organizations need — have become an important economic and cultural phenomenon. Sourceforge.net, a major infrastructure provider and repository for such projects, lists more than 10,000 of them and more than 300,000 registered users. The digital software products emanating from such projects are commercially attractive and widely used in business and government (by IBM, NASA and the German government, to name just a few). Because such products are deemed a "public good" — meaning that one person's use of them does not diminish another's benefits from them — the open-source movement's unique development practices are challenging the traditional views of how innovation *should* work.

A Brief History

In the 1960s and 1970s, software development was carried out mostly by scientists and engineers working in academic, government and corporate laboratories. They considered it a normal part of their research culture to freely exchange, modify and build upon one another's software, both individually and collaboratively. In 1969, the U.S. Defense Advanced Research Projects Agency (DARPA) established the ARPAnet, the first transcontinental, high-speed computer network. ARPAnet allowed developers to exchange software code and other information widely, easily, swiftly and cheaply. It grew in popularity and eventually linked several universities, defense contractors and research laboratories. However, its limits soon became apparent. The network could connect approximately 250 hosts, too few to cater to the growing communication needs among engineers and academics. A number of technological advancements that emerged between 1940 and 1970 led to the development of the Internet project that would eventually solve this bottleneck. Today the Internet has more than 100 million users worldwide and has become the major breeding ground for open-source software development.

The communal culture was strongly present among a group of programmers at the MIT Artificial Intelligence Laboratory in the 1960s and 1970s. In the 1980s, this group received a major jolt when MIT licensed some of the code to a commercial software firm, which promptly restricted access to the source code of that software, and hence prevented noncompany personnel — including MIT hackers who had participated in developing it — from continuing to use it as a platform for further learning and development.

Richard Stallman, an accomplished programmer at the Artificial Intelligence Laboratory, was rather distraught and somewhat offended by this loss of access to communally developed code, and he lamented a general trend in the software world towards the development of proprietary packages that could not be studied or modified by others. In 1985, he founded the Free Software Foundation with the intention to develop and diffuse a legal mechanism that would allow developers to preserve the "free" status of their software by using their own copyright to grant software licenses that would guarantee a number of rights to *all* future users. The basic license developed by Stallman, in order to implement this idea, was the General Public License or GPL. The basic rights transferred to those possessing a copy of free software included the right to use it at no cost, the right to study and modify its "source code," and the right to distribute modified or unmodified versions to others at no cost.

The free software idea did not immediately become mainstream; industry was actually rather suspicious of it. For example, firms feared the possible "viral effects" of the GPL license, meaning that, after software in the GPL regime is combined with proprietary software, it would prove difficult to restrict user access through normal licensing or controlling the source code. In 1998, Bruce Perens and Eric Raymond agreed that a significant part of the problem resided in Stallman's term "free" software. The term might understandably have an ominous ring to it when heard by individuals in the business world. Accordingly, they, along with other prominent hackers, founded the "open source" software movement. Open-source software incorporates essentially the same licensing practices as those pioneered by the free software movement, covering free redistribution of software and the inclusion of the source code of a program. These licensing

***Professor Georg von Krogh** is director of the Institute of Management, and cofounder of the research center KnowledgeSource, at the University of St. Gallen in Switzerland. Contact him at georg.vonkrogh@unisg.ch.*

practices also apply to derived works in that the rights attached to the original program apply to all who build upon the source code, without these programmers needing to provide additional licenses.

Incentives To Innovate

Under the aegis of open-source licensing practices, which guarantee that the products cannot be withheld from anyone's use, what are the incentives to innovate, and, given that most open-source projects exist outside the firm's boundaries, how does this innovation process work?

People and firms innovate because there are private incentives to do so. For example, entrepreneurs use their own funds to develop knowledge and products that generate revenue streams (and employees are paid for their creative services to the company). Society also encourages innovation by putting in place mechanisms to protect intellectual property associated with products, so that future revenue streams can be guaranteed for the innovator.

However, a central question raised by the success of open-source software development has been succinctly stated by two economists, Josh Lerner and Jean Tirole: "Why should thousands of top-notch programmers contribute freely to the provision of a public good?" Open-source software developers are rarely paid for their services, and the licenses and hacker practices make it difficult, if not impossible, for these developers to appropriate returns from their products. Eric von Hippel and I suggest that open-source software developers freely reveal and share because they garner personal benefits from doing so, such as learning to develop complex software, perfecting expertise with a computer language, enhancing their reputation, and for pure fun and enjoyment. Many of these benefits depend on membership in a well-functioning developer community. Typically, in open-source communities, members give direct, specific and immediate feedback on the software code that others write and submit. This peer-review process is not only valuable for the individual who

Referenced Research

J. Lerner and J. Tirole, "The Simple Economics of Open Source" NBER working paper no. w7600 (Cambridge, Massachusetts: National Bureau of Economic Research, March 2000)
A foundational paper for research on open-source software, arguing that individuals contribute to open-source software in accordance with career incentives.

E. von Hippel and G. von Krogh, "Exploring the Open Source Software Phenomenon: Issues for Organization Science" *Organization Science* 14, no. 2 (2003, in press)
The authors suggest that a combination of private and community-related benefits results from contributions to open-source software development projects.

G. Hertel, S. Niedner, and S. Herrmann, "Motivation of Software Developers in Open Source Projects: An Internet-Based Survey of Contributors to the Linux Kernel" *Research Policy* 32 (2003, in press)
The authors test two extant models from the social sciences. The first explains incentives to participate in social movements, and the second deals with motivational processes in small work teams, particularly "virtual teams." The authors report a good fit between models and data derived from a survey of 141 contributors to the Linux kernel.

N. Franke and E. von Hippel, "Satisfying Heterogeneous User Needs via Innovation Toolkits: The Case of Apache Security Software" *Research Policy* 32 (2003, in press)
The authors explore a frequently cited reason for contributing to open-source software: People innovate to better satisfy their own needs.

S. O'Mahony, "Guarding the Commons: How Open-Source Contributors Protect Their Work" *Research Policy* 32 (2003, in press)
An ethnographic study in which the author explores the various ways open-source project members encourage compliance with the terms of their project licenses.

E. von Hippel, "Economics of Product Development by Users: The Impact of 'Sticky' Local Information" *Management Science* 44 (May 1998): 629-644
Information about user needs and problems is "sticky," in the sense that it is costly to retrieve for a manufacturer. See also " 'Sticky Information' and the Locus of Problem Solving: Implications for Innovation," E. von Hippel, *Management Science* 40, no. 5 (1994): 429-439.

M.A. Cusumano, "Shifting Economies: From Craft Production to Flexible Systems and Software Factories" *Research Policy* 21, no. 5 (1992): 453-480
The author discusses the organization of large-scale, commercial software innovation.

R.D. Austin, "The Effects of Time Pressure on Quality in Software Development: An Agency Model" *Information Systems Research* 12, no. 2 (2001): 195-207
The author explores in detail the relationship between the software developer and the firm.

K. Lakhani and E. von Hippel, "How Open Source Software Works: 'Free,' User-to-User Assistance" *Research Policy* 32 (2003, in press)
How users demonstrate satisfaction and obligation in assisting each other in resolving tasks related to use of Apache software. Also see **J.Y. Moon and L. Sproull, "Essence of Distributed Work: The Case of the Linux Kernel,"** First Monday 5, no. 11 (Nov. 2000).

G. von Krogh, S. Spaeth and K. Lakhani, "Community, Joining, and Specialization in Open-Source Software Innovation: A Case Study" *Research Policy* 32 (2003, in press)
A study of growth and specialization in a community of developers.

S. Koch and G. Schneider, "Effort, Co-operation, and Co-ordination in an Open Source Software Project: GNOME" *Information Systems Journal* 12, no. 1 (2002): 27-42
Research on open-source projects from a software engineering perspective. The data are used for a first attempt to estimate the total effort to be expended on a particular project.

B. Kogut and A. Metiu, "Open-Source Software Development and Distributed Innovation" *Oxford Review of Economic Policy* 17, no. 2 (2001): 248-264
The authors of this paper argue that open-source software development is a production model that exploits the distributed intelligence of participants in Internet communities.